OPTIMIZATION OF SWARM INTELLIGENCE BASED SOLUTIONS FOR UNIVERSITY COURSE TIMETABLING SYSTEM

A Synopsis submitted in the partial fulfilment of the requirement for the award of the degree of

Doctor of Philosophy
In
Physics and Computer Science

Submitted by

Manpreet Kaur

Under the Supervision of

Dr. Sanjay Saini

Prof. G.S. Tyagi Head, Dept. of Physics & Computer Sc. Faculty of Science	**Prof. Ravindra Kumar** Dean, Faculty of Science

DEPARTMENT OF PHYSICS AND COMPUTER SCIENCE
FACULTY OF SCIENCE
DAYALBAGH EDUCATIONAL INSTITUTE
DAYALBAGH, AGRA (UP) – 282005

Contents

1 Introduction 1
2 Dayalbagh Educational Institute's (DEI) Faculty-wise Course Timetabling System 1
3 Soft Computing Tools to Solve such Problems 2
3.1 Operational Researches (OR) based techniques 2
3.1.1 Graph Coloring (GC) theory based technique 2
3.1.2 Integer/Linear programming (IP/LP) method 2
3.1.3 Constraint Satisfaction(s) Programming (CSPs) 2
3.2 Metaheuristic Approaches 3
3.2.1 Population based metaheuristic approaches 3
3.2.2 Single solution based metaheuristic approaches 5
3.3 Intelligent novel approaches 6
3.3.1 Hybrid Approaches 6
3.3.2 Fuzzy theory based approaches 6
3.4 Distributed multi agent systems approach 6
4 Literature Review 7
5 Timetables of different Faculties of Dayalbagh Educational Institute (DEI) 15
5.1 Faculty of Commerce 16
5.2 Faculty of Science 16
5.3 Faculty of Social Science 16
5.4 Other Faculties 16
6 Motivation 16
7 Objectives 17
8 Suggested Platform 17

1 Introduction

University Course Timetabling Problem (UCTTP) is a hybrid optimization problem in the class of NP-hard problems (Babaei H., 2015). A Course timetabling problem consists of scheduling a sequence of lectures between teachers and students in a prefixed period of time, satisfying a set of constraints of various types. It is the weekly scheduling for all the lectures of a set of university courses, removing the overlaps of lectures of courses having common students. The primary objective in preparing timetable is obtaining conflict free schedule for each activity sharing the same resources.

In Wren (1996) timetabling is defined as:
"Timetabling is the allocation, subject to constraints, of given resources to objects being placed in space time, in such way as to satisfy as nearly as possible a set of desirable objectives."

In the task of creating a timetable, some constraints have to be satisfied. There are basically two types of constraints:

- Soft constraints
- Hard constraints.

Soft constraints are those if get violated in scheduling, the output is still valid, where as hard constraints are those which if get violated, the timetable is no longer valid. The hard constraints determine the feasibility of the timetable, and the soft constraints determine its quality. Solving the Timetabling problem (TTP) generally means finding such an assignment, so that no hard constraints are violated which means timetable is feasible, and the number of violations of soft constraints are as low as possible so that the feasible timetable is as good as possible. The hard constraints generally include:

- No participant can be in two different places at the same time.
- Only one event can take place in one place at given timeslot.
- Assigned place has to fulfil certain requirements of the event (size, features, etc).

The soft constraints are very much problem specific. They depend on the additional features of the places and events, additional requirements for the participants schedule, personal preferences, organizational issues, geographical location of the places etc.

2 Dayalbagh Educational Institute's (DEI) Faculty-wise Course Timetabling System

For instance, if we look at Dayalbagh Educational Insitute's timetabling problem then we will find that different faculties of DEI have different type of hard and soft constraints for developing a feasible timetable. Faculties have interconnected timetables as teachers from one faculty can be assigned courses of other faculties, Students have options to study courses running in other faculties along with their major courses etc. Faculties run in two shifts. In some faculties, groups of students are formed on basis of their gender and girl students' group courses are to be scheduled in morning shifts only. Lectures per week for one course are decided on basis of course credits. Few courses have fixed timings. Similarly, many more constraints need to be considered while generating timetable for any faculty of DEI. All these constraints make timetabling problem of DEI very much

complex to solve. It involves a large amount of manual effort and time to generate a feasible timetable.

3 Soft Computing Tools to Solve such Problems

Main approaches used to solving the UCTTP problem up to now are as follows:

3.1 Operational Researches (OR) based techniques

These techniques include Graph Coloring (GC) theory based technique, Integer/Linear programming (IP/LP) method and Constraint Satisfaction(s) Programming (CSPs)

3.1.1 Graph Coloring (GC) theory based technique

In the graph coloring theory based timetabling problem, the events, constraints (preferably hard constraints) and timeslots are considered as nodes, edges and colors respectively. The chromatic number is the least number of colors which are required to color the nodes and edges of the graph. Colored nodes are equivalent to the number of timeslots. So the events can be scheduled in a timeslot in one day using these colors. If two adjacent nodes have the same color, then time conflict will occur. Graph coloring theory could perform a straightforward implementation of the problem and generate conflict free timetables in the output. Since finding the required number of colors to color the nodes of the graph has NP-hard time, so this method does not have efficient performance, unless it is combined with metaheuristic approaches to increase the performance.

3.1.2 Integer/Linear programming (IP/LP) method

IP/LP method is a complete mathematical method. It is applied on basis of the structure and type of faculty. Solutions of the UCTTP problem using this method are different for different education institutes. The size of the institute has great influence on solving this problem so that it is difficult to obtain the optimal solutions at the end of performing an IP/LP method for UCTTP problem. This method is mostly used separately without combination with other approaches. However some constructive heuristics can be used within this method which facilitates the analysis of constraints.

3.1.3 Constraint Satisfaction(s) Programming (CSPs)

The constraint satisfaction programming (CSP) based method is a computing based system where a constraint could be defined as a limitation over a space of facilities. This method aims to find a set of consistent values where each of these values could be assigned to the values of variables and satisfy the predefined constraints. The final solution is to assign values to each variable where these assignments could satisfy the whole of given constraints (Lewis, 2006; Obit, 2010).

3.2 Metaheuristic Approaches

These techniques include Case Base Reasoning method (CBR), population based approaches and single solution based approaches.

3.2.1 Population based metaheuristic approaches

Population based methods includes the following algorithms: Evolutionary and Genetic Algorithms (GAs), Ant Colony Optimization (ACO), Memetic Algorithm (MA), Harmonic Search Algorithm (HAS), Particle Swarm Optimization (PSO), Artificial Bee colony Optimization (ABC), Intelligent Water Drops (IWD) etc.

3.2.1.1 Evolutionary (EAs) and Genetic Algorithms (GAs)

EAs are based on computing model and inspired from natural evolutionary mechanism. Charles Darwin proposed Evolutionary algorithms (EAs) in 1859. Evolutionary algorithms are based on the simplified biological model of evolution and natural selection. EAs are population based algorithms that process whole set of candidate solutions simultaneously. While solving a particular problem, an environment is created by the parameters of problem, in which potential solutions can evolve. This encourages the evolution of good solutions. EAs act on a population of possible solutions and include three steps: (1) selection, (2) regeneration and (3) replacement. In the selection phase: people with high fitness are selected to be parents for the next generation. In the regeneration phase: two crossover and mutation operators are performed on parents who were selected in the first phase and in the replacement phase: the people of initial population are replaced by the newly created people.

Genetic algorithms are a subset of EAs. GAs are based on the following steps: (1) to generate the initial population, (2) to evaluate the generated population by using the evaluation function, (3) to select some people as parents to crossover based on the obtained information from the evaluation functions, (4) to apply crossover operator to produce children, (5) to apply mutation operator for children, (6) to select parents and children to form the new population for the future generation and (7) if the termination condition is satisfied, the algorithm stops else it goes to second step and continues (Obit, 2010).

3.2.1.2 Ant Colony Optimization (ACO)

Ant colony optimization is a meta-heuristic algorithm that finds an optimal solution over a discrete search space. Ant colonies have a built-in capability of optimization. They use probabilistic rules based on local information to find shortest path between two points in environment. In early 1990's, Macro Dorigo and colleagues introduced ACO by observing the foraging behavior of ant colonies. When ants search for food, initially they explore the surrounding area in random manner. While moving, ant deposits a certain chemical on the ground known as Pheromone. After finding the food, ant returns to its nest by again depositing the pheromone over their trails. The quantity of pheromone will guide other ants to the food source. The path with higher amount of pheromone trails is probabilistically chosen by the ants. The main principal behind this interaction is called Stigmergy, or communication through the environment.

3.2.1.3 Memetic Algorithm (MA)

Memetic algorithms are believed to be successful because they combine the explorative search ability of re-combinative evolutionary algorithms and the exploitive search ability of local search methods. The term memetic algorithm was invented by Moscato from Richard Dawkin's term "meme". A meme can be considered as a unit of information which generates itself when the beliefs of people change. Memetic algorithms are a kind of evolutionary algorithms that integrate knowledge of the problem in the form of heuristics, approximate algorithms, local search techniques, specialised recombination operators, and truncated exact methods, etc. In memetic algorithm, the space of possible solutions is reduced to the local optimization of sub spaces which are one of the best advantages of this method. Memetic algorithm is a combination of genetic and hill climbing algorithm (Obit, 2010).

3.2.1.4 Particle Swarm Optimization (PSO)

After being inspired by the study of bird flocking behavior by biologist Frank Heppner, James Kennedy and Russell Eberhart developed PSO in 1995. PSO approach can be used solve the problems whose solutions can be represented as a point in an n-dimensional solution space. A population of individuals, referred to as particles, is randomly set into motion through the space. Particle Swarm has two primary operators:

i) Velocity update
ii) Position update.

During each generation, each particle is accelerated toward its previous best position and global best position. At each iteration, a new velocity value is calculated for each particle using the distance from its previous best position and the distance from the global best position. This new calculated velocity value is then used to calculate the new position of the particle in the search space. This process is repeated for a set number of times or until the termination condition is met. This extremely simple approach has been surprisingly effective across a variety of problem domains.

3.2.1.5 Artificial Bee Colony Optimization (ABC)

Honey bees have several complicated behaviours such as mating, breeding and foraging. These behaviours have been mimicked for several honey bee based algorithms developed for optimization. Some of these are Bees Algorithm (BA), Bee System (BS), and Bee Colony Optimization (BCO), Artificial Bee Colony (ABC). Bee System was first identified by Sato and Hagiwara in 1997 and Bee colony Optimization (BCO) was introduced by Lucic and Teodorovic in 2001. Artificial bee colony was proposed by Karabora in 2005. Honey bees collect nectar or pollen from areas around their hive. Colony of bees can extend itself over long distance and in multiple directions simultaneously. Bees communicate with each other by a mechanism called "Waggle Dance" which is the figure of digit "8" and the dancers produce a loud buzzing sound by moving their bodies and exchange information of rich food source's location which comprises of the direction of the food source with respect to sun and distance from the hive. It also indicates the quality of the food.

3.2.2 Single solution based metaheuristic approaches

This method is also one of the metaheuristic methods which uses a single solution to analyse the problem instead of using an initial population to solve an optimization problem, so that at first a single solution is selected based on some criteria. Single solution algorithms include Tabu Search Algorithm (TS), Variable Neighborhood Search (VNS), Randomized Iterative Improvement with Composite Neighboring algorithm (RIICN), Simulated Annealing (SA) and Great Deluge Algorithm (GD) etc.

3.2.2.1 Tabu Search Algorithm (TS)

Tabu search algorithm is one of the metaheuristic optimization algorithms which was first proposed by Glover (1989). At first, this algorithm starts to move from an initial result, and then it selects the best neighbour result among the neighbors of current result. If this result was not in the Tabu list, the algorithm moves toward the neighbour result, otherwise the algorithm will check a criterion called aspiration criterion. Based on aspiration criterion, if the neighbor result is better than the available best result found by now, then the algorithm will move toward that result, even if that result is in the Tabu list. After the movement of algorithm toward neighbor result, the Tabu list is updated; it means that the previous movement by which we have moved toward the neighbor result is placed in the Tabu list in order to avoid return back to that result and cycle creation. The duration in which the movements are within the Tabu list is determined by a parameter called Tabu Tenure. Movement from the current result to the neighbor result continues until reaching the termination condition.

3.2.2.2 Variable Neighborhood Search (VNS)

Systematic change of neighborhood within a possibly randomized local search algorithm yields a simple and effective metaheuristic for combinatorial and global optimization, called variable neighborhood search (VNS). It was proposed by Mladenovic, Hansen (1997). It explores distant neighborhoods of the current incumbent solution, and moves from there to a new one if and only if an improvement was made.

3.2.2.3 Simulated Annealing (SA)

This method is a local search inspired of heating solids in physic science. The Simulated Annealing (SA) is considered to be the oldest metaheuristic. It was first proposed by Kirkpatrick *et al.* (1983). It is a Monte Carlo approach to optimization and its origins are in statistical mechanics (Metropolis algorithm). The term *simulated annealing* derives from the roughly analogous physical process of heating and then slowly cooling a substance to obtain a strong crystalline structure. In simulation, a minima of the cost function corresponds to this ground state of the substance. The simulated annealing process lowers the temperature by slow stages until the system *freezes* and no further changes occur. At each temperature the simulation must proceed long enough for the system to reach a steady state or equilibrium. This is known as *thermalization*. The time required for thermalization is the decorrelation time; correlated microstates are eliminated. The sequence of temperatures and the number of iterations applied to thermalize the system at each temperature comprise an annealing schedule.

3.3 Intelligent novel approaches

These include hybrid approaches, artificial intelligence based approaches, fuzzy theory based approaches, Clustering Algorithm based approaches etc.

3.3.1 Hybrid Approaches

Hybrid approaches have become more attractive in solving NP-complete problems recently and represents better efficiency and performance. In these methods, useful and efficient features of other techniques and algorithms are applied together to solve problems such as timetabling problem. Since the mentioned methods have some weaknesses, so combination of these methods can eliminate the weakness of individual methods and leads to a superior hybrid approach; therefore, this hybrid method can be led in generation of solutions more efficient in comparison to the results of each of these methods, individually. For example, a hybrid approach, in the first phase the possible initial solution is created by using the constraint programming technique and in the second phase the simulated annealing method is used to improve the initial generated solution and in the third phases the hill climbing solution is applied to improve and modify more.

3.3.2 Fuzzy theory based approaches

Fuzzy logic is one of the main principles of Computational Intelligence proposed by Lotfi Zadeh in 1965. Most important feature of fuzzy logic is that it can face incompleteness. Fuzzy logic works with partial truth values lying between exact zero and exact one. It is generally used to produce approximate results. Fuzzy logics resemble with human decision making ability. There is a vast domain where this technique tends to apply such as Image Processing and decision making. The lack of decisiveness in objects, things and etc. represent the lack of accuracy, ambiguous and generally fuzziness. Now in fuzzy science the intermediary which does not exist within mathematic is released. In fact, fuzzy logic is a continuous logic inspired of the approximate argumentation of human being (Asmuni, 2008).There are many products available that are manufactured based on fuzzy logic such as Fuzzy washing machines, Fuzzy microwaves etc.

3.4 Distributed multi agent systems approach

An agent could observe and receive anything from its environment through sensors and then performs over environment through a driver. Agents are classified into different classes based on their application, such as, autonomous, intelligent, reactionable, proactive, learner, mobile, cooperative/communicative agents (Srinivasan et al., 2011). Therefore, agents must have a common language and a communication media to cooperate with each other where these two components are essential among two agents.

Multi agent systems have a more general concept and for all types of current systems, including multiple autonomous components are applied to the following features: (1) each agent has the ability of solving a problem incompletely, (2) in multi agent systems there is no general control system, (3) data is distributed and (4) computations are asynchronous. However, after stating the autonomous features, the multi-capacity features in agents can be explained as:(1) dividing tasks among a large number of agents which are modular making, flexibility, modifiable and extensible, (2) the knowledge released over different resources (agents) has the capability of being integrated to completion, (3) applications

require distributed computations by distributed multi agent systems for better support and (4) the agents technology provides the summary and result of distributed components technology (Srinivasan et al., 2011; Wangmaeteekul, 2011). The advantages of the multi agent approach are to resolve collisions directly, variability of strategic negotiation and present some real problems and events.

4 Literature Review

The first timetabling problem has been solved using graph coloring problem (Welsh & Powell, 1967). However, this method was not able to solve the problems in cases where there were pre-assigned sessions. The graph coloring method to modelling a timetabling problem by using a non-directional graph had been described by De Werra (1985). Here the aim is to color the graph by using a given number of colors where no adjacent vertices (nodes) have the same color. The resulted timetable should not have a conflict and it should be colored with the least number of colors. Selim (1988) introduced the idea of separated graph vertices to reduce the chromatic number of graphs and applied it to separate students. The separation of one vertex is similar to separation of students in one course.

In order to reduce the number of penalties and create high quality timetables compared with manual timetables, the method of coloring the edges of two part graph also has been conducted by Hafizah and Zaidah (2010). Dandashi and Al-Mouhamed (2010) performed scheduling of classrooms using the graph coloring method where vertices and edges represent common courses and students, respectively, and the aim was to present a VC⁄ heuristic approach in order to promote the uniform distribution of courses over colors and balance the number of courses for each timeslot over the existing rooms. Another hybrid approach to solve UCTTP problem using genetic coloring has been proposed by Asham, Soliman, and Ramadan (2011) to reduce the cost of finding the least number of required colors to color a graph.

Dimopoulou & Miliotis (2001) proposed general structure of IP/LP method in UCTTP problem. This formulation in IP/LP is performed based on different groupings in two classes on courses grouping and time intervals grouping. To deal with large scale timetabling problems, Aubin and Ferland (1989) developed a more general procedure. They have separated the timetabling problem into two sub problem which are called timetabling sub problem and grouping sub problem. They proposed a heuristic approach to solve this problem through simultaneous application of sub problems of timetabling and grouping, until a final solution is reached and improved as much as possible.

Daskalaki, Birbas, and Housos (2004) presented the Integer Programming (IP) method for solving the UCTTP problems where the aim is to allocate a set of courses among teachers and groups of students and also a set of weekly and daily timeslots pairs. Daskalaki and Birbas (2005) presented an IP based two-step relaxation method to generate efficient solutions of timetabling in two steps. In first step, the sessions of courses which should be

held consecutively have been scheduled through allocation of courses to days and special times. During second step, it should be ensured that the courses (presented to the same group of students) require more than one timeslots have been scheduled consecutively. To organize courses and teachers, according to available timeslots and classrooms, an integer programming approach has been presented by Bakir and Aksop (2008), which results in reduction of dissatisfaction of students and teachers during the implementation of finite rules by a set of constraints simultaneously.

Deris, Omatu, Ohta, and Saada (1999) presented a possible and near to optimum solution to the course timetabling problem. They combined a genetic algorithm with constraint based reasoning. Deris, Omatu, and Ohta (2000) used a timetabling planning problem using the constraint based reasoning technique in an object oriented approach. Zhang and Lau (2005) applied ILOC software to implement the CSP approach to build timetabling in university.

Khonggamnerd and Innet (2009) used a genetic algorithm to sort a university timetable where the crossover rate was 70% and however, no hard constraint has been violated in timetabling. Used constraints result in more occupation of rooms and capacity of the rooms. Shaikh et al. (2016) presented Genetic Algorithm with meta- heuristic approach for generating timetable. Abdelhalim (2016) used a genetic algorithm that uses some heuristics to generate an initial population of feasible good quality timetables. A crossover type focusing on the utilization rates of learning spaces is introduced. A targeted mutation operator that uses a local search heuristic is also employed. The algorithm applies a composite fitness function that considers space utilization, gaps between events and a maximum number of lectures per day.

By using a learner machine, Alsmadi, Abo-Hammour, Abu-Al-Nadi, and Algsoon (2011) proposed a novel GA to solve the UCTTP problem. The results of this technique were to minimize the number of violated soft constraints, maximize the use of available rooms and reduce of teachers' workload. Sani, Yabo (2016) also presented GA to solve UCTTP.

Socha et al. (2002) used Max–Min ant system to generate university courses timetabling which has led to building an optimal path where each path could generate a constructive graph to allocate courses to timeslots affected by the amount of pheromone within a range. Application of ant colony optimization algorithm by Mayer, Nothegger, Chwatal, and Raidl (2008, 2012) for the UCTTP problem has been post enrolled according to ITC-2007 dataset where ants allocate events to rooms and timeslots based on two types of pheromone. This algorithm has good performance and leads to good results with high run time.

Applying a hybrid ant colony system has been proposed by Ayob and Jaradat (2009) to solve the UCTTP problem. Here, two types of hybrid ant system, including a combination of simulated annealing with ant colony and tabu search with ant colony have been presented. A number of ants perform the complete allocation of courses to timeslots based

on a predefined list. The selection of timeslots probabilities by ants to allocate courses has been done by using heuristic information and information of an indirect coordinator mechanism among agents and activities within an environment. Patrick, Godswill (2016) presented Greedy Ants Colony Optimization Strategy for Solving the Curriculum Based University Course Timetabling Problem.

The memetic algorithm has been implemented by Jat and Shengxiang (2008) to solve the UCTTP problem through a combination of local search method in genetic algorithms. One of the local searches has been run over the events and another one over timeslots.

Chen and shih (2013) applied particle swarm optimization (PSO) towards solving course timetabling problems. To reduce the computational complexity, a timeslot was designated in a particle's encoding as the scheduling unit. Two types of PSO, the inertia weight version and constriction version, were evaluated. Moreover, an interchange heuristic was utilized to explore the neighboring solution space to improve solution quality. Montero et al. (2011) applied PSO with local search procedure to handle the dynamism and to find high quality solutions.

Fong et al. (2015) proposed a hybrid swarm based approach using a variant of artificial bee colony algorithm which utilizes a global best model inspired from particle swarm optimization to enhance the global exploration ability while hybridizing with the great deluge algorithm in order to improve the local exploitation ability.

Alvarez, Crespo, and Tamarit (2002) applied Tabu search algorithm to assign students to the course groups in order to generate timetables with high quality and balance the number of students who registered in whole group and good results have been obtained. The allocation process in their method has two phases: (1) the first phase, to generate a set of solutions for one student and (2) the second phase, a combination of a set of solutions and applying TS with local strategies to obtain the high quality timetables without considering the worst solution(s) for each student and (3) the third phase, to allocate room and improve the allocation, of course without changing the initial assignment of courses to the timeslots. In the second phase, TS is used to improve the quality of initial timetable, also some moves are also used which are: (1) simple move, (2) exchange and (3) multiple exchanges where the influence of these moves are highlighted in this method. Hertz (1991) presented Tabu search for large scale timetabling problems. The objective was to reduce the number of conflicts due to courses taking place simultaneously but involving common students or teachers, or requiring the same classroom. In addition to these classical constraints they also take into account the grouping of students.

Aladag, Hocaoglu, and Basaran (2009) presented the influence of neighboring structures on a TS algorithm to solve the UCTTP problem where the influence of simple and swap moves is tested on TS operations which are based on neighboring structures. Two new neighboring structures have been proposed by using simple and swap moves. Here, four applied neighboring structures and the comparison of obtained results from these

structures has been specified. TS algorithm includes using advanced strategies and common components like Tabu list, various memories, neighboring structures and etc. One of the main factors is the influence of algorithm efficiency based on the defined neighboring structures which depends on the nature of problems. Here, two neighboring structures based on different types of moves like simple, swap and two other neighboring structures have been used which have been applied by Aladag and Hocaoglu in 2007 and Alvarez et al. (2002), respectively. The aim is to combine the various effects in the applied TS algorithm. Due to the obtained results, multiple comparisons have been presented among whole neighboring structure statistically.

Tuga, Berretta, and Mendes (2007) presented the combination of Kempe neighboring chain in the simulated annealing algorithm to solve the UCTTP problem. In this method, the hard constraints are reformulated by relaxation. So these constraints are created in the form a relaxed soft constraint. However, the relaxation problem is analyzed in two steps: (1) to create a feasible solution based on a heuristic based graph and (2) a simulated annealing algorithm has been used to minimize the violations of soft constraints. In the second phase, a Kempe neighboring chain based heuristic has been used.

The simulated annealing approach is presented by Aycan and Ayav (2008) to solve the UCTTP problem. They compared the efficiency of different neighboring search algorithms based on simple search, swap search, simple swap search and computed the run time cost for each method. The highest satisfaction of timetabling is obtained by a combination of three mentioned algorithms.

Joudaki, Imani, and Mazhari (2010) used local search algorithm and MA method to solve the UCTTP problem. These researchers presented a method based on an improved combination of SA and MA. In their research, they applied SA algorithm as a local search routine which increases the ability of extracting from MA. Although, the modification of the crossover operator in MA and creation of initial population through a heuristic based method was done in this algorithm. The optimization was performed by optimizing the generated chromosomes and minimizing the number of violations from the constraints and it was added as a new operator to MA. The success factors of MA were to use the heuristic capability of evolutionary algorithms and combined this advantage with the extraction ability of local search procedures.

Shengxiang and Jat (2011) used local and guided search strategy within the GA process to solve the UCTTP problem. The aim was to maximize the allocations and to minimize the violation of soft constraints.

The variable neighborhood search algorithm (VNS) was presented by Abdullah, Burke, and McColloum (2005) to solve the UCTTP problem which proposed a basic VNS and then to use an exponential Monte Carlo acceptance criterion by each solution. The main idea was to apply a Monte Carlo acceptance criterion for improvement of the explorations through acceptance of the best solution with the given probability in order to find the

number of promised neighbors. Abdullah, Burke, & McColloum, 2007a, 2007b presented a randomized iterative improvement with a composite neighboring algorithm (RIICN) to improve its previously presented algorithm which was in fact the combination of VNS and local search. The Tabu list was applied to the penalty of inefficient and un-promised neighboring structures after a given number of iterations.

A hybrid algorithm by Kostuch (2005) used sequential heuristic and simulated annealing to solve the UCTTP problem over ITC-2002 dataset. This method includes three phases as: using a sequential heuristic to generate the feasible initial timetables, applying simulated annealing to minimize the number of violations of soft constraints and also used simulated annealing algorithm to increase the improvement of generated timetables. Abdullah et al. (2007a, 2007b) presented a hybrid evolutionary approach to solve the UCTTP problem where the combinations of local search algorithm with evolutionary approaches give better results. The aim of problem was to completely meet of hard constraints and minimize the violations of soft constraints. The obtained results of a hybrid evolutionary approach showed the minimization of penalty values from soft constraints.

Abdullah and Hamdan (2008) also presented another hybrid approach to solve UCTTP problem implemented in three phases. The first phase was to generate the initial solution by using constructive heuristics, the second phase used an improved technique applying the randomized iterative improvement algorithm and the third phase also used simulated annealing as an acceptance criterion. However, the hill climbing was used in the third phase to promote the quality of timetables. Integration of two Tabu search and memetic algorithms to solve UCTTP problem was done by Turabieh and Abdullah (2009) where the crossover and mutation operators were used to select a solution from a population and then the random neighboring structures was reused for each solution whose quality of solutions were not promoted within Tabu list. The Tabu list was applied to penalty neighboring structures which did not have the capability to generate better solutions. Badoni, Gupta (2015) presented a hybrid approach to solve UCTTP which combines the merits of genetic algorithm and iterated local search algorithm for its convergence to global optima, at the same time avoiding being get trapped into local optima.

A multi population hybrid genetic algorithm was proposed by Shahvali Kohshori and Saniee Abadeh (2012) to solve UCTTP problem based on three genetic algorithms, FGARI, FGASA and FGATS. In this algorithm, fuzzy logic was used to evaluate the number of violations of soft constraints in the fitness function to deal with real world data which were ambiguous and non-deterministic and random, local search, simulated annealing and Tabu search methods would also be useful accompanied with fuzzy method to improve inductive search to satisfy the search ability.

To optimal fuzzy classification of students, Amintoosi and Haddadnia (2005) used a fuzzy function to solve UCTTP genetic programming problem. The aim was to separate the students of populous classes. This separation led to reduce the amount of conflict of students' courses in weekly program. Here, at first the fuzzy c-mean clustering algorithm

divide students into c-classes and then, according to the criteria of distance of clusters' centres, density of each cluster, co-entrance of students of each cluster and dimension ratio of clusters by using a fuzzy function, the value of clustering is determined so that by selecting the appropriate features (courses), the best classification of students is obtained.

A hybrid fuzzy evolutionary algorithm was presented by Rachmawati and Srinivasan (2005) to multi objective resource allocation problem which was a student's project allocation problem. Here, the student project allocation must satisfy a number of soft purposes in a sequence of some points. This algorithm used a fuzzy inductive system to model and collect purposes. Fuzzy system considered some priorities to decide on an agreement among the different purposes by which the direction of the search path toward attractive regions within purpose space was performed.

Asmuni, Burke, and Garibaldi (2005) presented a fuzzy multiple heuristic sorting method where the sorting of events was done through a simultaneous considering of three distinct heuristics by using fuzzy methods. The sequential combination of three heuristics was sorted as the highest degree, saturation degree and submission degree. The descending sorted events were allocated to the last timeslot with the least penalty cost sequentially while the feasibility was maintained during the whole process.

Golabpour, Mozdorani Shirazi, Farahi, kootiani, and beige (2008) presented a fuzzy solution based on memetic approach to solve university timetabling problem where a timetable has been compared with both genetic and memetic algorithms and its results may satisfy the existing constraints simultaneously during a shorter time interval. The aim was to use fuzzy logic as a means for local search in memetic algorithm. Chaudhuri and Kajal (2010) presented a fuzzy genetic heuristic idea to solve UCTTP problem where the genetic algorithm has been applied by using an indirect representation based on integration events, features and the fuzzy set model was also to evaluate the violation of soft constraints in objective function according to uncertainties of real world data. Here, a degree of uncertainty within objective function was considered for each soft constraint and this uncertainty was evaluated by formulating violation parameter from soft constraint in objective function using fuzzy membership functions.

Shahvali Kohshori, Saniee Abadeh, and Sajedi (2011) presented a fuzzy genetic algorithm accompanied with local search to solve UCTTP problem where the fuzzy genetic algorithm with a local search algorithm use inductive search to solve the combined problem and applied local search which has the ability of improving efficiency within genetic algorithm. The applied fuzzy logic within this approach was also used to evaluate the violation of soft constraints in objective function due to facing with uncertainty in real world data. Shatnawi, Al-Rababah, and Bani-Ismail (2010) used a novel clustering technique based on FP-Tree to solve UCTTP where the given technique was used to classify students based on their selective courses who submitted for the next semester. The aim of this clustering was to solve scheduling of courses where in the previous semesters the submission of students in some courses due to simultaneous scheduling has been

prevented, while in this technique no conflict would happen over scheduling of exams since no two exams at the same time would be taken for courses by two identical groups of students.

Gaspero, Missaro, and Schaerf (2004) used distributed multi agent architecture to generate course timetables. Here, UCTTP problem includes a set of courses in fixed timeslots in a circulating week. UCTTP problem refers to only a set of university departments. Each department has an education program corresponding to particular rules, constraints and purposes based on their resources and resources do not have shared feature, unless the resource exchange among departments would be useful which is done through negotiation. To solve the problem, a multi agent scheduling system based on one market with artificial money has been considered and each department includes three colleague agents: (1) to search a local solution, (2) to negotiate for resources with other departments and (3) to manage the related information.

In Kaplansky and Meisels (2004), a distributed timetabling system was in three software layers as: scheduling agents, using a negotiation protocol to generate a pervasive university scheduling and presenting network infrastructures. Protocol implementation was to negotiate among scheduling agents and classroom agent. The main attempt was focused on studying the various places in the protocol of negotiation among scheduling agents.

The studying of distributed timetabling problem based on scheduling agents by Meisels and Kaplansky (2003), almost in real timetabling problems include organized parts which required creating timetables for people included in an independent way, while some global constraints were considered.

Pedroso (2003) presented a multi agent system to create automatic timetabling with shared resources where an automatic timetabling system was proposed for a normal state of the universities. Agents (departments) compete for a set of classrooms over a number of given timeslots. Each agent applies their own algorithm and this algorithm may be unknown for other departments. A central system was assumed to determine whether some agents are allowed to allocate resources or not, which were also based on a list of requests received from each agent. The initial priority was evaluated by a number of attendances and some of the requirements are valued for particular features over resources.

To solve UCTTP problem, a global solution model based on agent was presented by Yanga, Paranjape, and Benedicenti (2006) where each agent within the system was a constraint in UCTTP problem. The agent based solutions include generality, flexibility, dynamism and scalability which act better than other modes. Each constraint agent was an independent module in the proposed model and new constraint agents had the ability to join and leave simply the timetabling system which would be the dynamism result of constraint in resolving through system technique and design.

One test was presented by Yanga, Paranjape, and Benedicenti (2004) to evaluate a system with mobile agents in UCTTP. Most focus was on the application of new technology of mobile agents to implement the solution of a general UCTTP problem. The mobile agents' technology was established based on TEEMA (TR Labs Execution Environment for Mobile) platform.

The study of UCTTP problem as distributed DisTTP (Distributed TimeTabling Problem) was done by Xiang and Zhang (2008). To solve DisTTP problem based on multiple segmented constraint networks (MSCNs), an alternative method was presented here. The network topology was based on sparse matrix. The proposed method maintained a part of timetabling from all particular agents. The distributed timetabling avoids translation and interpretation of concentrated local constraints and their communication where this process keeps the shorter scheduling and privatization of departments. The obtained solution of a multi agent system consists of an SA (Scheduling Agent) at each department and a CA (Central Agent).

The implementation of class timetabling has also been done by Nandhini and Kanmani (2009) based on multi agent systems where the implementation process has been presented through applying hill climbing algorithm with the sharpest upward slope (until reaching the ancestor). CombinationGenerator and MinFinders agents are applied to generate maximum input combinations and create a combination with the minimum evaluation function to consecutive exams, respectively. Using this proposed method would continue the initial random solution until reaching the given optimal solution.

A multi agent system was proposed by Oprea (2007) for UCTTP problem scheduling where two basic features were defined by the distribution and the dynamism of environment. An efficient solution to solve this problem could be provided by an agent based approach. The focus was on architecture of multi agent systems which is presented for UCTTP timetabling called MAS-UP-UCT. The advantage of this method includes a large number of communications, collaboration and negotiation among agents.

Yanga, Paranjape, Benedicenti, and Reedc (2006) presented a system model using mobile agents where a multi agent system was applied to generate the solutions of UCTTP problem. Four types of agents collaborate sequentially with each other to perform courses scheduling process as: course (mobile), signboard, publisher and mediator. The powerful key of this approach is to use the independency feature of agents and this independency was embedded clearly in the performance of course agent. Each course agent in the system was responsible to negotiate with other course's agents to find the satisfactory resource class (timeslots and rooms) in order to present and represent courses.

Solving UCTTP problems by using multi agent systems require the development of an intelligent decision making system proposed by Yanga and Paranjape (2011). UCTTP problem is a dynamic distribution problem which requires a system of decision making, where agent are independent and a flexible communication methodology has been used to

create the backbone of decision making system. Strnad and Guid (2007) have presented a new architecture for multi agent systems in solving UCTTP problem where agents allocate the required technical and human resources through negotiation as agencies from each course. The negotiation and description protocol of independency decisions of agents has been defined in the given framework format.

5 Timetables of different Faculties of Dayalbagh Educational Institute (DEI)

The goal of the university course timetabling problem is to find a method to allocate whole events to fix predefined timeslots and rooms, where all constraints within the problem must be satisfied. Events include students, teachers and courses where resources encompass the facilities and equipment's of classrooms such as theoretical and practical rooms. The first definition of timetabling has been presented as three sets of: (1) teachers, (2) classrooms and (3) timeslots (Gotlib, 1963). Most of the timetable system produced can only be used at the university involved in the research, as needs and requirements vary from one institution to another. It is very difficult to formulate a general model that is applicable for all cases, since every educational institution has its own special constraints and objectives. The constraints in the problem can vary and there may be many different objectives

- **Complexities**

The timetable construction problem is NP-complete (Cooper, 1995). The problem is, firstly, due to ever increasing number of students and secondly the number of constraints in each faculty being different. So the main difficulty is how to satisfy the number of soft constraints in UCTTP problem. The significance of this problem is mainly due to the difficulty of constructing a feasible timetable that satisfies the preferences of the administration, the instructors, and the students.

- **Constraints**

A feasible timetable is one in which all events have been assigned a timeslot and a room so that the following hard constraints are satisfied:

- The classes of two courses with the same student enrolment cannot be assigned to the same time slot.
- An instructor cannot teach more than one classes at the same time slot.
- The assigned number of classes that are scheduled for the same time slot cannot exceed the number of available classrooms.
- All class meetings should be assigned to a time slot.
- The room is big enough for all the attending students and satisfies all the features required by the event.
- Only one class takes place in any room at a given time.
- No class should be scheduled in lunch hour.
- No instructor should be assigned with more than 2 contiguous lectures.
- Lectures per week are decided on basis of course credits.

While generating timetables for various faculties of DEI, various constraints have to be followed such as:

5.1 Faculty of Commerce

- Inter-Faculty teachers i.e. some teachers come from other faculties to teach inter-faculty half courses. Schedule for these instructors must not clash.
- Student of some courses are divided into sections and different sections can be taught by same or different teachers.
- As faculty runs in two shifts (morning/evening), teachers must be available in both the shifts.
- Non-Faculty courses have fixed timing and these must be scheduled at 2:50 P.M.
- Core courses must be scheduled in afternoon classes only.
- Work-experience courses must have 4 lectures per week.

5.2 Faculty of Science

- Inter-Faculty teachers i.e. some teachers come from other faculties to teach inter-faculty half courses.
- Classes of Honours programs must be scheduled in morning shift only.
- Practical courses must be scheduled in continuous lectures.
- Half courses must be scheduled at 2:35 P.M.
- Core courses must be scheduled in first three days of the week.
- Continuous lectures must be assigned for work-experience courses.

5.3 Faculty of Social Science

- Courses of same group combination cannot be scheduled at same time.
- Inter-Faculty teachers i.e. some teachers come from other faculties to teach inter-faculty half courses. Schedule for these instructors must not clash.
- Students can opt course combinations from other faculties also e.g. Major Courses of Social Science+ Arts + Science faculties. Courses having same student must not be scheduled at same timeslot.
- Work-experience courses must be scheduled in continuous classes.
- Timings are fixed for Core-courses and Non-Faculty half courses.

5.4 Other Faculties

There are many other constraints that need to be considered while generating timetables for faculties of DEI such as continuous lecture should not be assigned to the teacher if related courses are from different faculties, outside faculty room availability is required to be considered for some courses, course scheduling of girl students must be done in morning shift for faculty of engineering, one teacher should not have more than two theory lectures in continuation etc.

6 Motivation

The complexity of the problem's hard constraints is one of the prime motivator in our study. The timetable is put together almost completely manually and there are many complex constraints. This results in most of the scheduler's energy being spent trying to create a

schedule that does not violate any of the hard constraints and where all students have access to the courses in their curriculum.

Solving a real world university timetabling problem manually often requires a large amount of time and expensive resources. In order to handle the complexities of the problems and to provide automated support for human timetable makers, much research in this area has been carried out over the last few years. This creates an interest in developing algorithms to tackle this important class of problems in a more effective way than what currently exists and investigate how to improve the quality of the timetable solution.

7 Objectives

The prime objective is to find a timetable solution in which least number of constraints are violated, minimize the length of the total time period over which the courses are to be scheduled and to find a feasible solution subject to a fixed total time period and several other constraints specific to a given scenario.

It is proposed to:

- Come up with an algorithm to reach a first cut solution to timetable for a given set of hard and soft constraints for a particular scenario.
- Development of modified swarm intelligence based algorithms, such as Ant Colony Optimization (ACO) and Intelligent Water Drops algorithms (IWD), for the multi-objective optimization of the obtained first cut solution of the timetables, having various constraints and objectives such as Teacher's time span and student's time span.
- Undertake extensive studies to come up with the best algorithm and its implementation after exhaustive comparisons of our solutions. Also to compare and contrast our solutions with existing solutions to some standard problems.
- Test the applicability of these algorithms on the timetabling of various Faculties of DEI.
- Construction of an interface for automatic development, manipulation and further re-organization of the timetable.

8 Suggested Platform

We will use MATLAB for initial testing of the newly developed algorithm. Visual Basic and MySql database is being proposed to be used for final implementation.

References

1. Abdelhalim et al. (2016). A Utilization-based Genetic Algorithm for Solving the University Timetabling Problem (UGA). *Alexandria Engineering Journal*, 55, 1395–1409.
2. Abdullah (2006). Heuristic approaches for university timetabling problems, Ph.D. Thesis, School of Computer Science University of Nottingham.
3. Abdullah, Burke, McColloum (2005). An Investigation of Variable Neighborhood Search for University Course Timetabling. 2nd multidisciplinary conference on scheduling: Theory and applications, 413–427.
4. Abdullah, Hamdan (2008). A hybrid approach for university course timetabling. *IJCSNS International Journal of Computer Science and Network Security*, 8(8).

5. Aladag, Hocaoglu, Basaran (2009). The effect of neighborhood structures on tabu search algorithm in solving course timetabling problem. *Expert Systems with Application*, 36, 12349–12356.
6. Aldeeb et al. (2015). Solving University Examination Timetabling Problem Using Intelligent Water Drops Algorithm. *International Parallel Conferences on Researches in Industrial and Applied Sciences Panel I: Engineering Applications*, 18- 30.
7. Alkan, Özcan (2003). Memetic Algorithms for Timetabling. *Proceedings of IEEE Congress on evolutionary Computation*, 1796 – 1802.
8. Alsmadi et al. (2011). A novel genetic algorithm technique for solving university course timetabling problems. *7th International Workshop on Systems, Signal Processing and their Applications IEEE*. 195-198.
9. Alvarez, Crespo, Tamarit (2002). Design and implementation of a course scheduling system using Tabu search. *European Journal of Operational Research*, 137, 512–523.
10. Amintoosi, Haddadnia (2005). Fuzzy C-means clustering algorithm to group students in a course into smaller sections. Berlin Heidelberg: Springer-Verlag,47–160.
11. Asham, Soliman, Ramadan (2011). Trans genetic coloring approach for timetabling problem. Artificial intelligence techniques novel approaches & practical applications, IJCA, 17–25.
12. Asmuni (2008). Fuzzy methodologies for automated university timetabling solution construction and evaluation, Ph.D. Thesis, School of Computer Science University of Nottingham.
13. Asmuni, Burke, Garibaldi (2005). Fuzzy multiple heuristic ordering for course timetabling. *In The proceedings of the 5th United Kingdom workshop on computational intelligence (UKCI05)*, 302–309.
14. Aycan, Ayav (2009). Solving the course scheduling problem using simulated annealing. *Advance Computing Conference, IACC*. IEEE International.
15. Ayob, Jaradat (2009). Hybrid Ant Colony Systems for Course Timetabling Problems. *2nd Conference on Data Mining and Optimization*. 120-126.
16. Babaei et al. (2015). A survey of approaches for university course timetabling problem. Computers & Industrial Engineering (86) 43–59.
17. Babaei, Hadidi (2014). A Review of Distributed Multi-Agent Systems Approach to Solve University Course Timetabling Problem. Advances in Computer Science: an International Journal, 3(5),19-28.
18. Badoni, Gupta (2015). A hybrid algorithm for university course timetabling problem. *Innovative Systems Design and Engineering*, 6(2), 60-66.
19. Bakir, Aksop,(2008). A 0–1 integer programming approach to a university timetabling problem. *Hacettepe Journal of Mathematics and Statistics*, 37(1), 41–55.
20. Bonabeau et al. (1999). Swarm Intelligence: From Natural to Artificial System. *Oxford University Press, New York*.
21. Burke, Petrovic (2002). Recent research directions in automated timetabling. *European Journal of Operational Research*. 140, 266–280.
22. Carter (2000). A Comprehensive Course Timetabling and Student Scheduling System at the University of Waterloo. *Practice and Theory of Automated Timetabling III*, 2079, 64-82.
23. Chaudhuri,Kajal (2010). Fuzzy genetic heuristic for university course timetable problem. *International Journal of Advance Soft Computing and its Applications*, 2(1). ISSN 2074-8523.
24. Chen, Shih (2013). Solving University Course Timetabling Problems Using Constriction Particle Swarm Optimization with Local Search. *Algorithms*, 6,227-244.
25. Cooper, Kingston (1993). The solution of real instances of the timetabling problem. The Computer Journal, 36(7), 645-653.
26. Cruz-Chávez et al. (2016). Solving a Real Constraint Satisfaction Model for the University Course Timetabling Problem: A Case Study. *Mathematical Problems in Engineering*. DOI http://dx.doi.org/10.1155/2016/7194864
27. Daskalaki, Birbas (2005). Efficient solutions for a university timetabling problem through integer programming. *European Journal of Operational Research*, 160(2005), 106–120.
28. De Werra (1985). An Introduction to Timetabling. European Journal of Operational Research, 19, 151–162.
29. Deb (2005). Multiobjective Optimization Using Evolutionary Algorithms. Chichester: Jhon Wiley & SonsLtd.
30. Deris, Omatu, Ohta (2000). Timetable planning using the constraint based reasoning. *Computers & Operations Research*, 27, 819–840.
31. Dimopoulou, Miliotis (2001). Theory and Methodology Implementation of a university course and examination timetabling system. European Journal of Operational Research, 130, 202–213.
32. Dorigo, Stützle. (2004). *Ant Colony Optimization*. A Bradford Book. The MIT Press.

33. Doulaty, Derakhshi, Abdi (2013). Timetabling: A State-of-the-Art Evolutionary Approach. *International Journal of Machine Learning and Computing*, 3(3). 255-258.
34. Eiben, Smith, James (2015). *Introduction to Evolutionary Computing*. Springer.
35. Fong et al. (2015). A Hybrid Swarm Based Approach to University Timetabling. *IEEE Transactions on Evolutionary Computation*, DOI 10.1109/TEVC.2015.2411741.
36. Gaspero, Missaro, Schaerf (2004). A multi-agent architecture for distributed course timetabling. *In proceedings of the 5th international conference on the practice and theory of automated timetabling (PATAT '04)*, 471–474.
37. Glover (1989). Tabu Search-Part I ORSA Journal on Computing 1(3),190-206.
38. Golabpour et al.. (2008). A fuzzy solution based on Memetic algorithms for timetabling. *IEEE international conference on multimedia and information technology*, 108–110.
39. Hafizah, Zaidah,(2010). Bipartite graph edge coloring approach to course timetabling. IEEE, 229–234.
40. Hertz (1991). Tabu search for large scale timetabling problems. *European Journal of Operational Research*. 54, 39-47.
41. Irene et al. (2008). A Study on PSO-based University Course Timetabling Problem. *IEEE International Conference on Advanced Computer Control*. 648-650.
42. Jat, Shengxiang (2008). A memetic algorithm for the university course timetabling problem. *In IEEE 20th IEEE international conference on tools with artificial intelligence*, 427–433.
43. Jat, Yang (2008). A Memetic Algorithm for the University Course Timetabling Problem. 20th IEEE International Conference on Tools with Artificial Intelligence. 427-433.
44. Joudaki, Imani, Mazhari (2010). Using improved Memetic algorithm and local search to solve University Course Timetabling Problem (UCTTP).
45. Kaplansky, Kendall, Meisels, Hussin (2004). Distributed examination timetabling. *In Proceeding of the 5th international conference on the practice and theory of automated timetabling (PATAT '04)*, 511–516.
46. Kennedy, Eberhart (1995). Particle Swarm Optimization. *Proceedings of the IEEE International Conference on Neural Networks, Perth*. 1942-1945.
47. Khonggamnerd, Innet (2009). On improvement of effectiveness in automatic university timetabling arrangement with applied genetic algorithm. IEEE. DOI 10.1109/ICCIT.2009.202
48. Kingston (2014). Timetable construction: the algorithms and complexity perspective. *Annals of Operations Research*, 218(1), 249-259.
49. Kirkpatrick *et al.* (1983). Optimization by Simulated Annealing. Science, New Series, 220(4598), 671-680.
50. Kostuch (2005). The university course timetabling problem with a three-phase approach. *Lecture Notes in Computer science*. Berlin/Heidelberg: Springer, 109–125.
51. Kumar,Singh, Sharma (2013). Automated timetable generator using particle swarm optimization. *International Journal on Recent and Innovation Trends in Computing and Communication*, 1(9), 686-692.
52. Lewis (2006). Metaheuristics for university course timetabling. Ph.D. Thesis,Napier University.
53. Mayer, Nothegger, Chwatal, Raidl (2008). Solving the post enrolment course timetabling problem by ant colony optimization. In Proceedings of the 7th international conference on the practice and theory of automated timetabling.
54. Mladenovic, Hansen (1997). Variable Neighborhood Search. *Computers and Operations Research* 24(11),1097-1100.
55. Montero et al. (2011). A PSO algorithm to solve a Real Course+Exam Timetabling Problem. *International conference on swarm intelligence*, 24, 1-8.
56. Nandhini, Kanmani (2009). Implementation of class timetabling using multi agents. IEEE. DOI: 10.1109/IAMA.2009.5228065
57. Nothegger et al. (2012). Solving the post enrolment course timetabling problem by ant colony optimization. *Annals of Operations Research*, 194, 325-339.
58. Obit (2010). Developing novel meta-heuristic, hyper-heuristic and cooperative search for course timetabling problems. Ph.D. Thesis, School of Computer Science, University of Nottingham.
59. Oprea (2007). MAS_UP-UCT: A multi-agent system for university course timetable scheduling. *International Journal of Computers, Communications & Control*, II(1), 94–102.
60. Osman (2015). Particle Swarm Optimization Based Najran University Course Timetable Scheduling. *NNGT Int. J. on Artificial Intelligence*, 2, 1-10.
61. Pandey, Sharma (2016). Survey on University timetabling problem. *3rd International Conference on Computing for Sustainable Global Development*, 160-164.

62. Patrick, Godswill (2016). Greedy Ants Colony Optimization Strategy for Solving the Curriculum Based University Course Timetabling Problem. *British Journal of Mathematics & Computer Science*, 14(2), 1-10.
63. Pillay (2016). Evolving construction heuristics for the curriculum based university course timetabling problem. *IEEE Congress on Evolutionary Computation*. 4437 – 4443.
64. Rachmawati, Srinivasan (2005). A hybrid fuzzy evolutionary algorithm for a multi-objective resource allocation problem. *In IEEE Proceedings of the fifth international conference on hybrid intelligent systems.*
65. Sani, Yabo (2016). Solving Timetabling problems using Genetic Algorithm Technique. *International Journal of Computer Applications,*134(15), 33-38.
66. Schaerf (1999). A Survey of Automated Timetabling. *Artificial Intelligence Review*. 13(3), 87–127. Springer.
67. Selim (1988). Split vertices in vertex colouring and their application in developing a solution to the faculty timetable problem. The Computer Journal, 31(1), 76–82.
68. Shah-Hosseini (2009). The intelligent water drops algorithm: a nature-inspired swarm-based optimization algorithm. Int. J. Bio-Inspired Computation, 1(1/2), 71–79.
69. Shahvali Kohshori, Saniee Abadeh (2012). Hybrid genetic algorithms for university course timetabling. *International Journal of Computer Science Issues*, 9(2), 2.
70. Shaikh et al. (2016). Genetic Algorithm With Meta- Heuristic Approach For Generating Timetable. *International Journal of Advanced Research*, 4(2), 300-304.
71. Shatnawi, Al-Rababah, Bani-Ismail (2010). Applying a novel clustering technique based on FP-tree to university timetabling problem: A case study. IEEE. DOI: 10.1109/ICCES.2010.5674875
72. Shengxiang, Jat (2011). Genetic algorithms with guided and local search strategies for university course timetabling. *IEEE Transactions on Systems, MAN, and Cybernetics-PART C: Applications and Reviews*, 41, 1.
73. Socha et al. (2003). Ant Algorithms for the University Course Timetabling Problem with Regard to the State-of-the-Art. *Applications of Evolutionary Computing*, 2611, 334-345.
74. Socha, Knowles, Samples (2002). A max-min ant system for the university course timetabling problem. *In Proceedings of the 3rd international workshop on ant algorithms (ANTS 2002).* Lecturer notes in computer science, 2463, 1–13. Springer-Verlag.
75. Srinivasan et al. (2011). Multi-agent based decision support system using data mining and case based reasoning. *IJCSI International Journal of Computer Science Issues*, 8(4), 2.
76. Strnad, Guid (2007). A multi-agent system for university course timetabling. *Applied Artificial Intelligence: An International Journal*, 21(2), 137–153.
77. Tuga, Berretta, Mendes (2007). A hybrid simulated annealing with kempe chain neighborhood for the university timetabling problem. *In 6th IEEE/ACIS international conference on computer and information science*.
78. Turabieh, Abdullah (2009). Incorporating Tabu search into memetic approach for enrolment-based course timetabling problems. *IEEE 2nd conference on data mining and optimization*, 115–119.
79. Wangmaeteekul (2011). Using distributed agents to create university course timetables addressing essential desirable constraints and fair allocation of resources. Ph.D. Thesis, School of Engineering & Computing Sciences Durham University.
80. Welsh & Powell (1967). An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal*, 10, 85–86.
81. Wren (1996). Scheduling, timetabling and rostering – A special relationship? *Practice and Theory of Automated Timetabling* (1153) 46–75.
82. Xiang, Zhang (2008). Distributed university timetabling with multiply sectioned constraint networks. *In Proceedings of the twenty-first international FLAIRS conference*.
83. Yanga, Paranjape (2011). A multi-agent system for course timetabling. *Intelligent Decision Technologies Computer Science and Artificial Intelligence*, 5(2), 113–131. IOS Press.
84. Yanga, Paranjape, Benedicenti, Reedc (2006). A system model for university course timetabling using mobile agents. *Multi-agent and Grid Systems – An International Journal*, 2, 267–275. IOS Press.
85. Zhang, Lau (2005). Constructing university timetable using constraint satisfaction programming approach. *In IEEE Proceedings of the 2005 International conference on computational intelligence for modeling, control and automation, and international conference on intelligent agents, web technologies and internet commerce* (CIMCA-IAWTIC'05), 2 , 55–60.